RIACS

Research Institute for Advanced Computer Science
NASA Ames Research Center

IN-61
43093
P15

Queueing in Networks of Computers

Peter J. Denning

17 May 91

RIACS Technical Report TR-91.14

NASA Cooperative Agreement Number NCC 2-387

Queueing in Networks of Computers

Peter J. Denning

Research Institute for Advanced Computer Science
NASA Ames Research Center

RIACS Technical Report TR-91.14
17 May 91

The designers of networks of computers must assess the capacity of the network to complete work within reasonable times. The utilization law, Little's law, forced-flow law, and response time formula are simple tools that can be used to calculate throughput and response times of networks. Bottleneck analysis can be used to calculate simple lower bounds on response time in terms of individual server parameters and the load on the network as a whole. These simple results are important tools for all users of scientific networks -- "back of the envelope" calculations can quickly reveal the effects of distant servers on local throughput and response time.

This is a preprint of the column *The Science of Computing* for
American Scientist 79, No. 3 (May-June 1991).

Work reported herein was supported in part by Cooperative Agreement NCC 2-387
between the National Aeronautics and Space Administration (NASA)
and the Universities Space Research Association (USRA).

Queueing in Networks of Computers

Peter J. Denning

Research Institute for Advanced Computer Science

17 May 91

A major airline has set up a transaction system used by its ticket agents to sell seats on its aircraft. The airline has authorized 1,000 agents around the country to make reservations from their workstations. There is a "disk farm" -- a large collection of magnetic-disk storage devices -- in New York that contains all the records of flights, routes, and reservations. On average, each agent issues a transaction against this database once every 60 seconds. One of the disks contains a directory that is consulted on every transaction to locate other disks containing the actual data; on average, each transaction by an agent accesses the directory disk 10 times. The directory disk takes an average of five milliseconds to service each request and it is busy 80 percent of the time.

How many transactions per hour are serviced nationwide on this system? What is the average response time experienced by an agent in Los Angeles? What would happen to response time if a new method of storing the directory reduced accesses to five per transaction? What would happen to response time if the number of agents were doubled?

These are typical questions relating to the capacity of a network of computers to complete the work requested of it. Most people think that the answers cannot be calculated without detailed knowledge of the system structure -- the locations and types of the agents' workstations, the communication bandwidth between each workstation and the disk farm, the number and types of disks in the farm, access patterns for the disks, local processors within the farm, amount of random-access memory in the farm, the type of operating system, the types of transactions that can be issued, and more. It may come as a surprise, therefore, that the first two questions -- concerning throughtput and response time -- can be answered precisely from the information given. For the changes of configuration proposed in the third and fourth questions, reasonable estimates of system behavior can be made from the available information and a few plausible assumptions.

Servers and Transactions

A computer network is composed of a number of interconnected servers. Servers include workstations, disks, processors, databases, printers, displays, and any other devices that can carry out computational tasks. Each server receives and queues up messages from other servers specifying tasks of the type that the receiving server is designed to carry out -- a typical message might ask a server to run a computationally intensive program, to perform an input/output transaction, or to access a database. A transaction is a specified sequence of tasks submitted to the network; when a server completes a particular task, it deletes the request from its queue and sends a message to another server, requesting that it perform the next task in the same transaction.

Measurements of servers are always conducted during a definite observation period. Basic measures typically include event counters and timers. These and other measures derived from them are called operational quantities. Invariant relations among operational quantities that hold in every observation period are called operational laws.

By counting outgoing messages and by measuring the time that the queue is nonempty, it is easy to measure the output rate X, the mean service time S, and the utilization U of a server. These three empirical quantities satisfy the relation $U=SX$, known as the Utilization Law (*Figure 1*). Similarly, by measuring the "space-time" accumulated by queued tasks, it is easy to determine the mean queue length $\overline{n}$ and the mean response time R; these quantities satisfy the relation $\overline{n}=RX$, known as Little's Law (*Figure 2*).

The utilization law and Little's law are counterparts of well-known limit theorems for stochastic queueing systems in a steady state. These theorems will usually be verified in actual measurements, not because steady state has been attained, but because the measured quantities obey the operational laws (*1, 2*).

The tasks making up a transaction can be regarded as a sequence of visits by the transaction to the servers of the network. The average number of visits per transaction to a particular server i is called the visit ratio V_i for that server; the server's output rate X_i and the system's output rate X_0 satisfy the relation $X_i=V_iX_0$, known as the forced-flow law (*Figure 3*). This remarkable law shows that knowledge of the visit ratios and the output rate of any one server is sufficient to determine the output rates of every other server and of the system itself. Moreover, any two networks with the same visit ratios have the same flows no matter what is the interconnection structure among their servers.

In a network, a server's output is a portion of the input to another server or of the output of the system. It simplifies an analysis to assume that the input and output flows of a server are identical and can be called the throughput -- a condition known as flow balance. The definitions do not imply flow balance. In most real systems there is a bound on the number of tasks that can be in the system at once; as long as the number of completions at every server is large compared to this bound, the error introduced by assuming flow balance will be negligible. For this reason, flow balance does not generally introduce much error for practical systems.

When a network of servers receives all its requests from a finite population of N users who each delay an average of Z seconds each until submitting a new transaction, the response time for a request in the network satisfies the response-time formula $R = N/X_0 - Z$ (*Figure 3*). This formula is exact for flow balance.

These formulas are sufficient to answer the the throughput and response time questions posed earlier for the airline reservation network. We are given that each transaction generates an average of 10 directory-disk requests, and so $V_i=10$ for the server represented by the directory disk. The mean service time at the directory disk is five milliseconds, so that $S_i=0.005$ seconds. The directory disk's utilization is 80 percent: $U_i=0.8$. Combining the forced-flow law and utilization law, we have for total system throughput:

$$X_0 = X_i/V_i = U_i/V_i S_i = 0.8/(10\times0.005) = 16 \text{ transactions per second}$$

Thus the entire airline reservation system is processing 57,600 transactions per hour. The response time experienced by any one of the 1,000 agents is

$$R = N/X_0 - Z = 1000/16 - 60 = 2.5 \text{ seconds}$$

Changing the Configuration

Consider next the two configuration questions. They ask for grounded speculations about response time in a future measurement period having different conditions -- for example, the directory disk visit ratio is reduced, or the number of agents is increased. Since operational laws deal only with relations among quantities observed in a past measurement period, they are not sufficient for making predictions. We must introduce additional, forecasting assumptions, that extrapolate measured parameter values from the past observation period into the future observation period; the laws can then be used to calculate the response time expected in that future period.

One common type of forecasting assumption is that, unless otherwise specified, the demands placed by transactions on servers, V_i, will be the same in the future period as they were in the measurement period. Similarly, unless otherwise specified, the mean service times, S_i, which depend primarily on mechanical and electrical properties of devices, will be the same. The utilizations, throughputs, and response times will change when any of these parameters changes.

The first configuration question asks what happens if the directory search strategy is changed so that transactions make only five accesses to the directory disk, half the previous number. Now the paucity of information about the parameters of the entire network limits the accuracy of the answer. There are two extremes. One possibility is that some disk other than the directory disk is the bottleneck of the system: most of the transactions are queued there, and its utilization is near 100 percent. Under these

conditions, reducing the demand for the directory disk will have a negligible effect on the utilization and throughput of the bottleneck disk; the forced-flow law tells us that the overall throughput and response time will therefore be unchanged.

At the other extreme, the directory disk is the bottleneck of the system; halving its demand will double system throughput. For the numbers given above, the response time formula yields a calculated response time of -28.75 seconds. The obvious absurdity of a negative response time -- signifying that answers are received before questions are asked -- indicates that the disk directory cannot be the bottleneck after demand on it is reduced by half, even if it were the bottleneck originally. All we can say with the given information and the given forecasting assumptions is that halving the demand for the directory disk will reduce the response time from 2.5 seconds to some smaller but still nonzero and nonnegative value. If the 2.5-second response time is acceptable, this proposed change of directory search strategy would not be cost effective.

Consider the second configuration question: what happens to the response time if the number of agents is doubled? Again, we are limited by the lack of knowledge of the other disks. If the directory disk is the bottleneck, then doubling the number of agents is likely to increase its utilization to 100%, giving a saturation value of throughput:

$$X_0 = 1/V_i S_i = 1/(10\times0.005) = 20 \text{ transactions per second}$$

With the response-time formula, these values yield:

$$R = N/X_0 - Z = 2000/20 - 60 = 40 \text{ seconds}$$

If the directory disk is not the bottleneck, some other server will have a smaller saturation

throughput, forcing response time to be longer than 40 seconds. Doubling the number of agents would produce unacceptably high response time.

This example illustrates that bottleneck analysis is a recurrent theme in forecasts of throughput and response time. Suppose that the visit ratios and mean service times are known for all the servers and do not vary with N. Each server generates a potential bottleneck that would limit the system throughput to $1/V_i S_i$, and would give a lower bound to the response time of $NV_i S_i - Z$. Obviously the server with the largest value of $V_i S_i$ gives the least upper bound on the throughput and is the real bottleneck. The products $V_i S_i$ are sufficient to determine lower bounds on the response time as a function of N (*Figure 5.*)

The operational laws coupled with bottleneck analysis offer a simple but powerful method for performance analysis. For systems whose visit ratios and service times do not vary with overall load, the products $V_i S_i$ -- the total service time requirement for each server -- are sufficient to answer these questions. The methods can be extended to yield efficient algorithms for computing throughput, response time, and mean queue length at every server as a function of the load N on the system (*1,2*).

Operational analysis is not a replacement for traditional queueing theory; it is a reinterpretation for the common case of measured data. Many of the steady-state limit theorems of queueing theory turn into operational laws or formulas that hold for flow-balanced networks.

The genesis of the operational interpretation was in the mid 1970s, when performance analysts were discovering that the formulas of Markovian queueing systems worked very well to predict utilizations, throughputs, and response times in real networks

of computers, even though the Markovian assumptions were grossly violated. Jeffrey P. Buzen proposed the operational hypothesis: many of the traditional steady-state queueing formulas are also relations among observable quantities under simple and general conditions (*1*). This hypothesis has been substantiated in practice and has become the underpinning for a large number of computer programs that calculate performance measures for networks of servers ranging from computers to manufacturing lines.

These simple results are important tools for all users of scientific networks -- "back of the envelope" calculations can quickly reveal the effects of distant servers on local throughput and response time.

References

1. Peter J. Denning and Jeffrey P. Buzen. 1978. "Operational analysis of queueing networks." ACM *Computing Surveys 10*, 3 (September). 225-261.
2. Edward D. Lazowska, John Zahorjan, G. Scott Graham, and Kenneth C. Sevcik. 1984. *Quantitative System Performance*. Prentice-Hall.

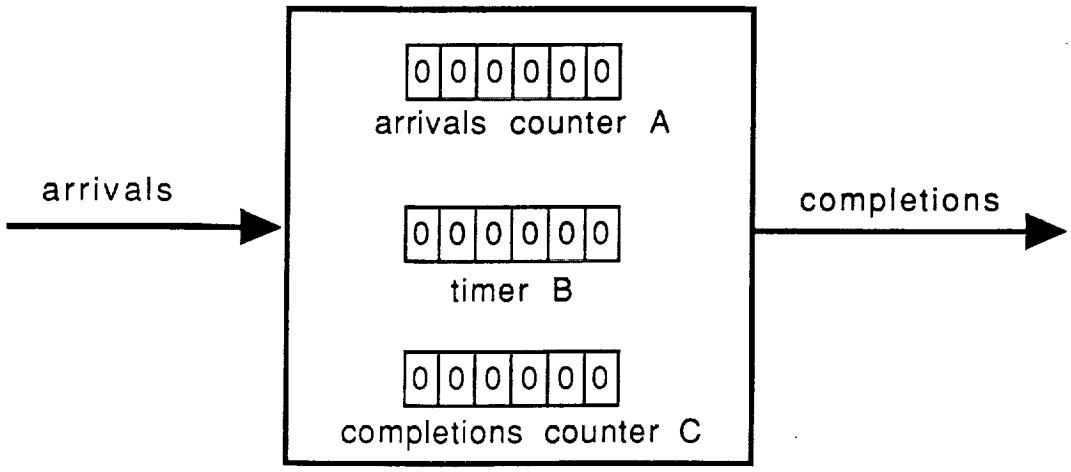

Figure 1. Task-processing server is the basic element of a network of computers. Over an observation period of length T, the counter A registers the number of arrivals, the counter C records the number of tasks completed, and the timer B measures the total busy time (the time when tasks are present). The utilization of the server is $U=B/T$, the output rate is $X=C/T$, and the mean service time per completed task is $S=B/C$. Because $B/T = (C/T)(B/C)$, we have the utilization law: $U=XS$.

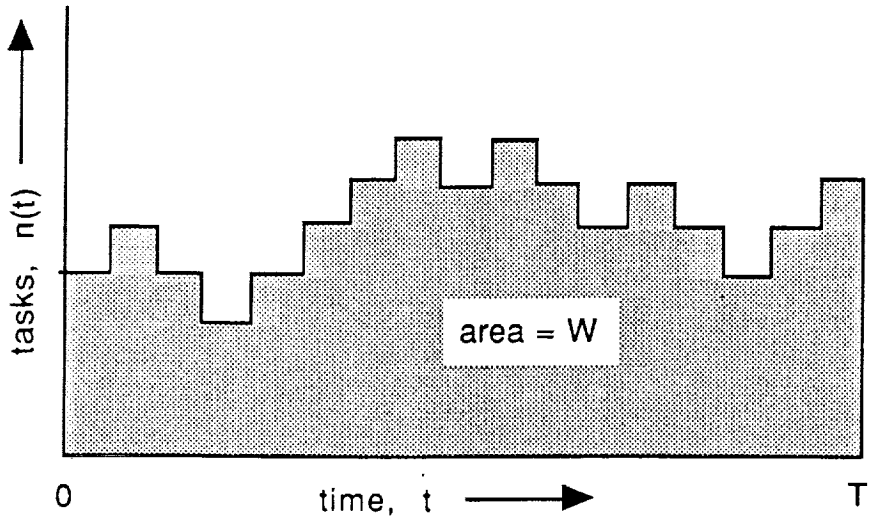

Figure 2. Average response time of a server can be calculated from just a few measurements of the system's performance. Let $n(t)$ denote the number of tasks in the server at time t. Let W denote the area under the graph of $n(t)$ in the interval from time 0 to time T; W is the number of task-seconds of accumulated waiting. The mean number of tasks at the server is $\bar{n}=W/T$, and the mean response time per completed task is $R=W/C$. Because $W/T = (C/T)(W/C)$, we have Little's Law: $\bar{n}=XR$. The mean service time S and the mean response time R are not the same; R includes queueing delay as well as service time.

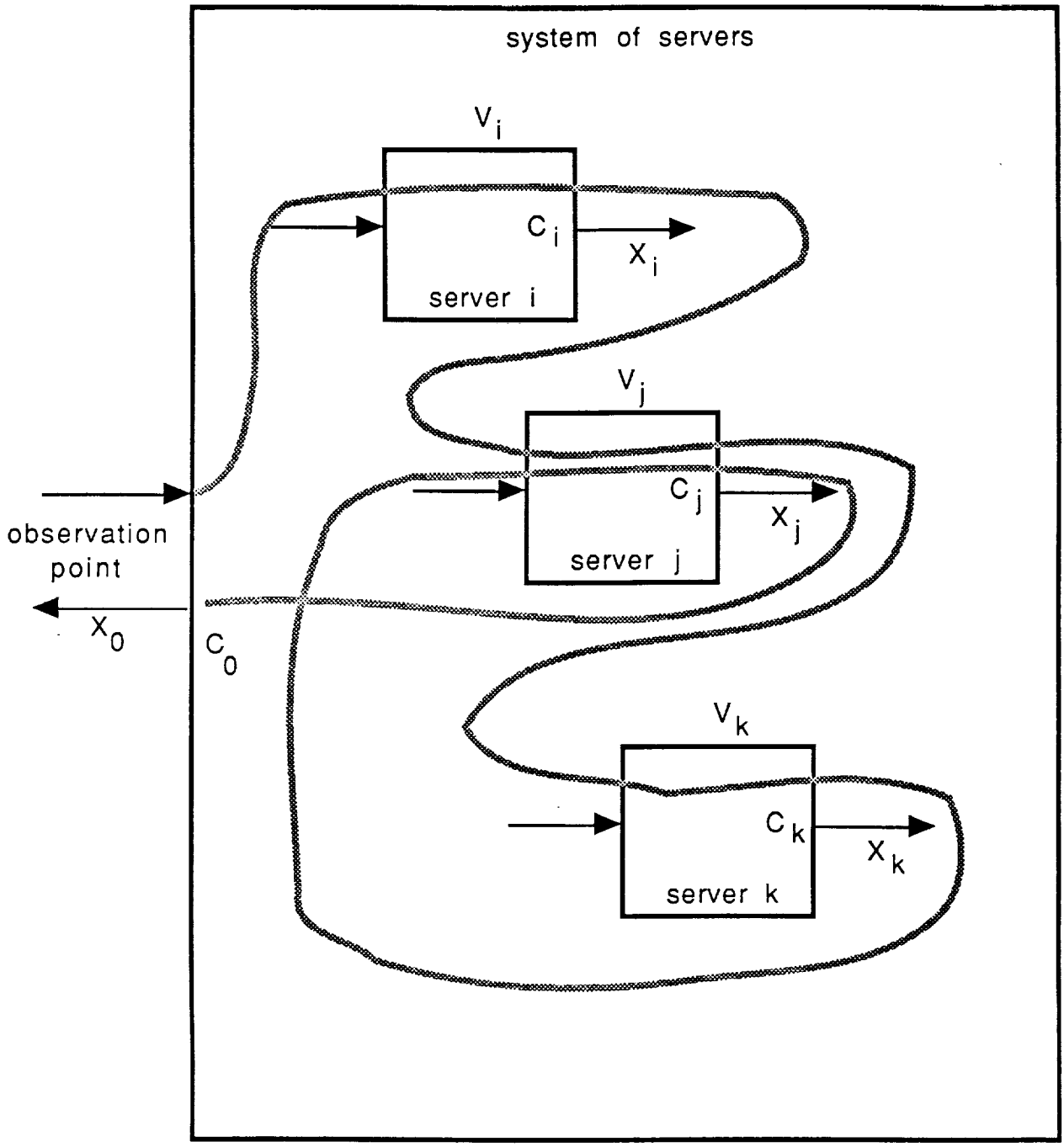

Figure 3. Flow of transactions through a network of servers can be calculated from a few selected measurements of performance. Here the transaction visits servers i and k once each and visits server j twice. Over an observation period, C_0 transactions are completed by the system. The average number of tasks per transaction for server i is $V_i = C_i/C_0$; V_i is called the visit ratio because each task is regarded as a "visit" by the transaction to the server. Because $C_i/T = (C_i/C_0)/(C_0/T)$, we have the forced-flow law: $X_i = V_i X_0$. This law says that the task flow at one point of the system determines the task flows everywhere. This law holds regardless of the interconnections among the servers; two networks with with the same visit ratios will have the same flows.

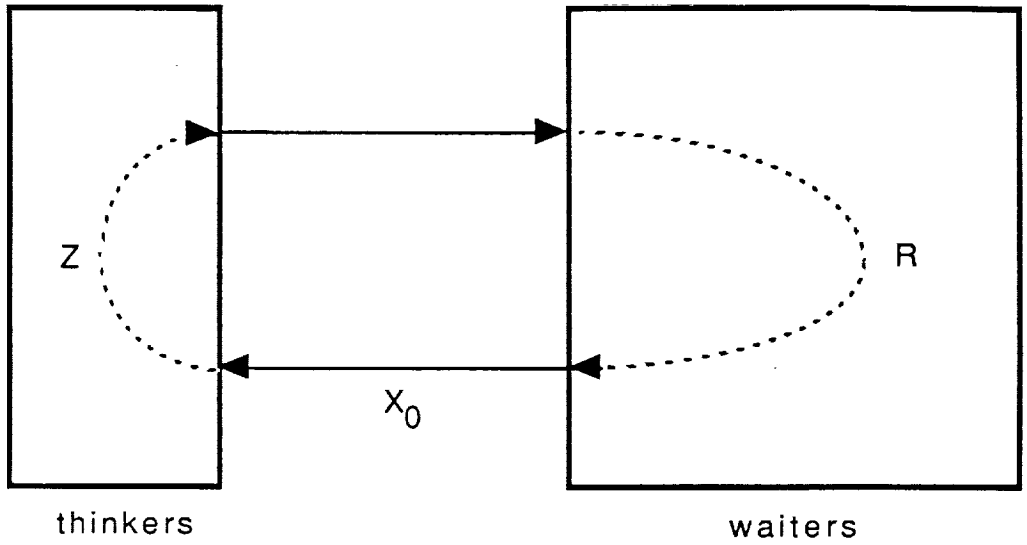

Figure 4. Users of a transaction system alternate between periods of "thinking" and periods of "waiting" for a response from the system. The total number of users -- thinkers and waiters -- is N. The average response time per transaction is R and the average thinking time is Z. Little's Law says says that the mean number of active users in an entire system is equal to themean response time of the system multiplied by the flow through the system. These three quantities are, respectively, N, $R+Z$, and X_0. Solving for the response time -- or in other words the average period spent waiting -- we obtain the response-time formula: $R = N/X_0 - Z$. Since the system includes a fixed number of thinking and waiting users, this formulation assumes one system arrival for each system completion (flow balance).

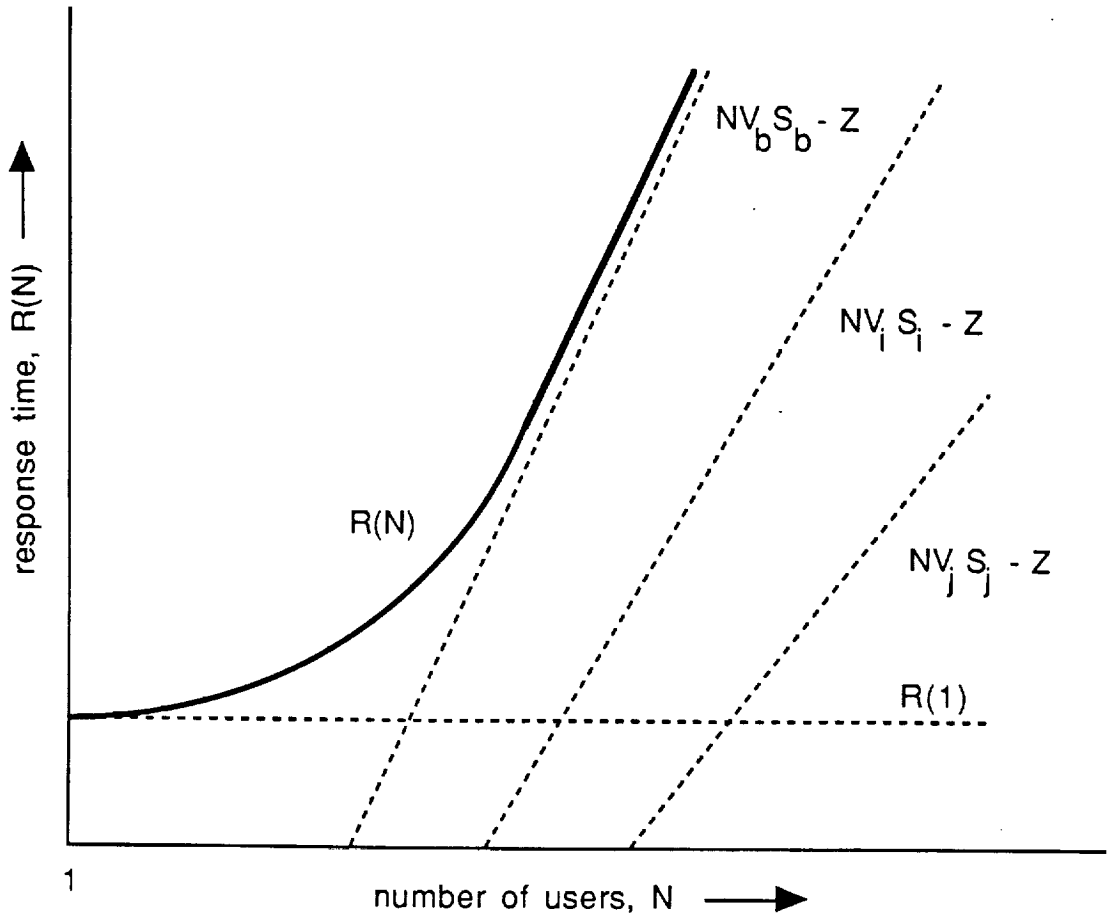

Figure 5. Bottleneck analysis shows how the response time changes as function of N. When N=1, the single user's jobs encounter no queueing delays from other jobs, whence $R(1) = V_1S_1+...+V_KS_K$, where K is the number of servers. Combining the utilization and forced-flow laws, $X_0 = X_i/V_i = U_i/V_iS_i \leq 1/V_iS_i$ since $U_i \leq 1$. Thus, $R(N) \geq NV_iS_i - Z$ for all i. Each of these lines is a potential asymptote for $R(N)$ with large N. The actual asymptote is determined by the largest of the potential asymptotes. Taking server b (for bottleneck) to be the one with largest V_iS_i, we have $R(N) \geq NV_bS_b - Z$. The bottleneck analysis assumes that the products V_iS_i do not vary with N.